📊 Optimitive Monthly Report Generator - Deployment Guide

🚀 Quick Start

1. Prerequisites

- Python 3.8 or higher
- Azure AD App Registration with SharePoint permissions
- SharePoint Online site access

2. Installation

```bash
# Clone or create project directory
mkdir optimitive-monthly-report
cd optimitive-monthly-report

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt

# Create Streamlit config directory
mkdir .streamlit
```

3. Configuration

A. Azure AD App Setup

1. Go to Azure Portal
2. Navigate to **Azure Active Directory** > **App registrations**
3. Click **New registration**
4. Configure:
   - Name: `Optimitive Monthly Report`
   - Supported account types: `Single tenant`
   - Redirect URI: Leave blank
5. After creation, note the:
   - **Application (client) ID**
   - **Directory (tenant) ID**

6. Go to **Certificates & secrets**:
   - New client secret
   - Description: `Monthly Report App`
   - Expires: Choose appropriate duration
   - **Copy the secret value immediately** (shown only once)
7. Go to **API permissions**:
   - Add permission > Microsoft Graph > Application permissions
   - Add these permissions:
     - `Sites.Read.All` (or `Sites.ReadWrite.All`)
     - `Files.Read.All` (or `Files.ReadWrite.All`)
   - Click **Grant admin consent**

B. Configure Secrets

1. Copy the template:

```bash
cp .streamlit/secrets.toml.template .streamlit/secrets.toml
```

2. Edit `.streamlit/secrets.toml` with your values:

```toml
[auth]
# Generate secure cookie key
cookie_key = "your-random-32-char-minimum-key"

# Generate password hashes
# python -c "import streamlit_authenticator as stauth; print(stauth.Hasher(['your_password']).generate())"
passwords = ["your-hashed-passwords-here"]

[graph]
tenant_id = "your-tenant-id"
client_id = "your-client-id"
client_secret = "your-client-secret"
hostname = "yourcompany.sharepoint.com"
site_path = "sites/YourSite"
```

4. Run the Application

Local Development

```bash
streamlit run app_monthly_report.py
```

Access at: http://localhost:8501

Production Deployment Options

Option A: Streamlit Cloud (Recommended)

1. Push code to GitHub (exclude `.streamlit/secrets.toml`)
2. Go to share.streamlit.io
3. Deploy from GitHub repository
4. Add secrets in Streamlit Cloud settings

Option B: Azure App Service

```bash
# Create requirements for Azure
echo "streamlit==1.28.0" > requirements.txt
echo "python-3.9" > runtime.txt

# Deploy using Azure CLI
az webapp up --name optimitive-monthly-report --resource-group your-rg --runtime "PYTHON:3.9"
```

Option C: Docker Container

```dockerfile
# Dockerfile
FROM python:3.9-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY . .
EXPOSE 8501
CMD ["streamlit", "run", "app_monthly_report.py", "--server.port=8501", "--server.address=0.0.0.0"]
```

```bash
# Build and run
docker build -t optimitive-report .
docker run -p 8501:8501 -v $(pwd)/.streamlit:/app/.streamlit optimitive-report
```

📁 Project Structure

```
optimitive-monthly-report/
├── app_monthly_report.py      # Main application
├── requirements.txt           # Python dependencies
├── .streamlit/
│   ├── config.toml           # Streamlit configuration (optional)
│   └── secrets.toml          # Secret configuration (DO NOT COMMIT)
├── .gitignore                # Git ignore file
├── README.md                 # Project documentation
└── DEPLOYMENT.md             # This file
```

🔐 Security Best Practices

1. **Never commit secrets.toml to version control**

   ```bash
   echo ".streamlit/secrets.toml" >> .gitignore
   ```

2. **Use strong passwords and rotate regularly**

   ```python
   # Generate strong password hashes
   import streamlit_authenticator as stauth
   import secrets

   # Generate random password
   password = secrets.token_urlsafe(16)
   print(f"Password: {password}")
   print(f"Hash: {stauth.Hasher([password]).generate()[0]}")
   ```

3. **Limit Azure AD permissions to minimum required**
   - Use `Read` permissions unless `Write` is needed
   - Regularly review and audit permissions

4. **Enable HTTPS in production**
   - Use SSL certificates
   - Configure secure headers

🧪 Testing

Basic Functionality Test

```python
```

```
# test_connection.py
import msal
from config import TENANT_ID, CLIENT_ID, CLIENT_SECRET

app = msal.ConfidentialClientApplication(
    client_id=CLIENT_ID,
    client_credential=CLIENT_SECRET,
    authority=f"https://login.microsoftonline.com/{TENANT_ID}"
)

result = app.acquire_token_for_client(scopes=["https://graph.microsoft.com/.default"])
if "access_token" in result:
    print("✅ Authentication successful")
else:
    print(f"❌ Authentication failed: {result.get('error_description')}")
```

SharePoint Connection Test

```
bash

# Run with test mode
streamlit run app_monthly_report.py -- --test
```

🔧 Troubleshooting

Common Issues

1. Authentication Errors

- **Issue**: "Invalid client secret"
- **Solution**: Regenerate client secret in Azure AD and update secrets.toml

2. Permission Errors

- **Issue**: "Insufficient privileges"
- **Solution**: Ensure admin consent is granted for Graph API permissions

3. SharePoint Access

- **Issue**: "Site not found"
- **Solution**: Verify site path format: `sites/SiteName` (no leading slash)

4. PDF Generation

- **Issue**: "WeasyPrint not available"

- **Solution**:

```bash
# Ubuntu/Debian
sudo apt-get install python3-cffi python3-brotli libpango-1.0-0 libpangoft2-1.0-0
pip install weasyprint

# macOS
brew install pango
pip install weasyprint

# Windows
# Download GTK installer from https://github.com/tschoonj/GTK-for-Windows-Runtime-Environment-Installer
pip install weasyprint
```

📊 Performance Optimization

1. Caching Configuration

```toml
# .streamlit/config.toml
[server]
maxUploadSize = 100
maxMessageSize = 100

[browser]
gatherUsageStats = false

[theme]
base = "dark"
primaryColor = "#E31E32"
```

2. Session State Management

- Use `st.cache_data` for expensive operations
- Implement pagination for large file lists
- Limit concurrent API calls

3. Resource Limits

```python
```

```python
# In app_monthly_report.py
MAX_FILE_SIZE_MB = 100
MAX_FILES_PER_BATCH = 50
API_TIMEOUT_SECONDS = 30
```

🧮 User Management

Adding New Users

```python
# generate_user.py
import streamlit_authenticator as stauth

# Generate password hash
username = "newuser"
password = "SecurePassword123!"
name = "New User"

hashed = stauth.Hasher([password]).generate()[0]

print(f"Add to secrets.toml:")
print(f'names = [..., "{name}"]')
print(f'usernames = [..., "{username}"]')
print(f'passwords = [..., "{hashed}"]')
```

Password Reset

1. Generate new password hash
2. Update `.streamlit/secrets.toml`
3. Restart application

🚀 Production Checklist

- [ ] Azure AD app configured with proper permissions
- [ ] Secrets.toml configured with production values
- [ ] Strong passwords and secure cookie key
- [ ] HTTPS enabled
- [ ] Error logging configured
- [ ] Backup strategy in place
- [ ] Monitoring setup
- [ ] User documentation provided
- [ ] Admin contact information updated

📞 Support

For issues or questions:

- **Developer**: Juan Cruz E.
- **Company**: Optimitive
- **Website**: optimitive.com
- **Documentation**: Check `/docs` folder for additional guides

📄 License

Last Updated: 2024